Claims

The following is a copy of Applicant's claims that identifies language being added with underlining ("___") and language being deleted with strikethrough ("—"), as is applicable:

1. (Currently amended) A method for facilitating profiling of an application, comprising:

intercepting an application ~~instructions~~ instruction immediately before ~~their~~ its execution;

determining if code associated with the intercepted application instruction has already been stored in a code cache of an interface layer;

if associated code has been stored in the code cache, executing the associated code from the code cache in lieu of the intercepted application instruction;

if the associated code has not already been stored in the code cache, determining if ~~an~~ the application instruction is a frequently used instruction according to a pre-established policy; ~~and~~

only if the application instruction is a frequently used instruction, instrumenting the application instruction so as to facilitate collection of information about execution of the application instruction; and

storing the instrumented application instruction in the code cache such that it will be executed in lieu of the intercepted application instruction when the application instruction is intercepted again.

2. (Currently amended) The method of claim 1, wherein the intercepted application ~~instructions are~~ instruction is an application ~~binaries~~ binary.

3. (Original) The method of claim 1, wherein determining if an application instruction is a frequently used instruction comprises consulting a program counter associated with the application instruction.

4. (Original) The method of claim 1, wherein instrumenting the application instruction comprises instrumenting the application instruction to collect information as to the fact that the application instruction was executed.

5. (Original) The method of claim 1, wherein instrumenting the application instruction comprises instrumenting the application instruction to collect information as to what other application instruction called the intercepted application instruction.

6. (Original) The method of claim 1, wherein instrumenting the application instruction comprises instrumenting the application instruction to collect information as to what application instructions the intercepted application instruction calls.

7. (Currently amended) The method of claim 1, wherein instrumenting the application instruction comprises instrumenting the application instruction to increment a counter representing a number of processor cycles ~~or a counter representing the number of instructions executed~~ required to execute the application instruction.

8. (Currently amended) The method of claim 1, further comprising recording information as to the execution of the intercepted application ~~instructions~~ instruction.

9. (Original) The method of claim 8, wherein recording information comprises recording information as to the execution of code stored in a shared library that the application accesses.

10. (Currently amended) The method of claim 1, further comprising determining if code associated with the intercepted application ~~instructions~~ instruction has been cached.

11. (Currently amended) The method of claim 10, further comprising executing the cached code in lieu of the intercepted application ~~instructions~~ instruction if associated code has been cached.

12. (Currently amended) A system for facilitating profiling of an application, comprising:

means for intercepting application instructions before they are executed;

means for determining if code associated with intercepted application instructions has already been stored in a code cache of an interface layer;

means for executing the associated code from the code cache in lieu of the intercepted application instructions;

means for determining if ~~an~~ application instructions are used frequently; ~~and~~

means for instrumenting frequently used application instructions to facilitate collection of information about execution of the application instructions; and

means for storing the instrumented application instructions in the code cache such that they will be executed in lieu of the intercepted application instructions when the application instructions are intercepted again.

13. (Currently amended) The system of claim 12, wherein the means for determining if ~~an~~ application instructions ~~is a~~ are frequently used instructions comprise means for counting the number of times the application instructions ~~is~~ are executed.

14. (Currently amended) The system of claim 12, wherein the means for instrumenting the application instructions comprise means for instrumenting the application instructions to collect information as to at least one of the fact that the application instructions ~~was~~ were executed, what other application instructions called the intercepted application instructions, and other application instructions the intercepted application instructions ~~calls~~ call.

15. (Currently amended) The system of claim 12, wherein the means for instrumenting the application instructions comprise means for instrumenting the application instructions to increment a counter representing a number of processor cycles required to execute the application instructions.

16. (Original) The system of claim 12, further comprising means for recording information as to the execution of the intercepted application instructions.

17. (Currently amended) A program that facilitates profiling of an application and that is stored on a computer-readable medium, the program comprising:

logic configured to intercept application binaries;

logic configured to determine if code associated with intercepted application binaries has already been stored in a code cache of an interface layer;

logic configured to execute the associated code from the code cache in lieu of the intercepted application binaries;

logic configured to determine if ~~an~~ application instructions ~~is a~~ are frequently used instructions; ~~and~~

logic configured to instrument the application instructions that are determined to be frequently used instructions so as to facilitate collection of information about execution of the application instruction; and

logic configured to store the instrumented application instructions in the code cache such that they will be executed in lieu of the intercepted application instructions when the application instructions are intercepted again.

18. (Currently amended) The program of claim 17, wherein the logic configured to determine if ~~an~~ application instructions ~~is a~~ are frequently used instructions comprises logic configured to count the number of times the application instructions ~~is~~ are executed.

19. (Currently amended) The program of claim 17, wherein the logic configured to instrument the application instructions comprises logic configured to instrument the application instructions to collect information as to at least one of the fact that the application instructions

~~was~~ were executed, what other application instructions called the intercepted application instructions, and other application instructions the intercepted applications instruction ~~calls~~ call.

20. (Currently amended) The program of claim 17, wherein the logic configured to instrument the application instructions comprises logic configured to instrument the application instructions to increment a counter representing a number of processor cycles.

21. (Currently amended) The program of claim 17, further comprising logic configured to record information as to the execution of the intercepted application instructions required to execute the application instructions.

22-27. (Canceled)

28. (Currently amended) A method for facilitating profiling of an application, comprising:

intercepting application code fragments prior to their execution;

determining if code associated with the fragments has already been ~~cached~~ stored in a code cache of an interface layer;

executing the ~~cached~~ associated code from the code cache in lieu of the intercepted code fragments if associated code has been ~~cached~~ stored;

determining ~~the number of~~ how many very long instruction words (VLIWs) ~~being~~ are executed for each code fragment;

instrumenting the application code fragments if they are determined to be frequently executed fragments; and

~~caching code associated with the code fragments that includes instrumentation~~ storing the instrumented application code fragments in the code cache such that they will be executed in lieu of the intercepted application code fragments when the application code fragments are intercepted again.

29. (Original) The method of claim 28, wherein instrumenting the code fragments comprises instrumenting the fragments to collect information as to the fact that the fragments were executed.

30. (Original) The method of claim 28, wherein instrumenting the code fragments comprises instrumenting the fragments to increment a counter representing a number of processor cycles.